CS112 - Fall 2022

Lab05

Instructor: Paul Haskell

INTRODUCTION

This week you will develop programs in which you create your own class and in which you explore the capabilities of String class. The programs you write will be a bit longer and trickier than those in earlier labs. A few of the programs will be adapted for use in the big projects we do later in the semester.

Reading command line arguments

First, something simple. As we discussed in lecture, the `args` variable in

```
static public void main(String[] args)
```

contains the "arguments" or parameters that are part of the command line when a program is called. For example,

```
%java ReadArgs.java Argument1 Arg2 3 FourthArgument
```

`args[]` is a list, with entries `args[0], args[1]`, … up to `args.length-1`. Your first program for this lab shall be called "**ReadArgs.java**" and it should output:

Program called with <<number of arguments>> arguments:

Then, your program should print each argument, with one argument per output line.

Be sure to test this program yourself, with different numbers of arguments. The TA's and I grade your programs with test software that does exactly that: runs your program with different numbers of arguments, collects the output, and checks if the output is correct.

String class capabilities

The String class offers many useful functions for modifying Strings, querying the contents of Strings, comparing Strings, etc. The online documentation for String (as well as other Java classes) can be found online simply by searching for "java String" in your favorite search engine.

- The docs.oracle.com site has official documentation
- Other high-ranked search results will give useful examples and tutorials

You will need to review this material for your next program, "**FixCapitalization.java**".

Book Editing

Some very famous authors have rejected standard rules of punctuation and capitalization. For example, an excerpt from William Faulkner's *The Sound and the Fury* reads:

> told me the bone would have to be broken again and inside me it began to say Ah Ah Ah and I began to sweat what do I care I know what a broken leg is all it is it wont be anything I'll just have to stay in the house a little longer that's all and my jaw-muscles getting numb and my mouth saying Wait Wait just a minute through the sweat ah ah ah behind my teeth and Father damn that horse damn that horse…

As assistant editor for a book publisher, you have been given a manuscript in which the author followed standard rules of punctuation but capitalized aT RAnDoM. It is your job to clean up the capitalization, and you will write a program called "**FixCapitalization.java**" to do it. The input to FixCapitalization.java is a single command-line argument. That argument is a String--a long string--with multiple words and with punctuation. The output will be the same text, but with capitalization corrected. You will use simple rules of capitalization, only requiring at the output that:

> the first word of every sentence be capitalized,
>
> no other capital letters exist in the output

Here are some hints for your program:

- Words are separated by spaces and possibly punctuation
- Sentences end with '.' or '!' or '?'
- The String.toLowerCase() method may be useful (and other String methods too)

As with the previous assignment, your first step is to think and sketch out in words how the program should work. When do you capitalize? When do you get rid of capitalization?

Be sure to test your program well! You can send multiple words to your java program as a single command-line argument if you put all the words in double-quotes, e.g.

> % java FixCapitalization.java "thIS iS A single cOMMand LINE arguMent. Yes, it is LoNG! "

Your Own Class

You did so well using Java in your first editing assignment that your boss at the book publisher gave you another job. This time, you are asked to write a program called "**Plagiarism.java**" that processes manuscripts to try to characterize who is the author. As before, the main `class Plagiarism` reads in a single command line argument with a long String. But inside the **Plagiarism.java** file, you will add your own class, called `WordCounter`. `class Plagiarism` feeds words from the input one-at-a-time to an object of type `class WordCounter`. The `WordCounter` counts:

- Total number of words
- Total number of times the word is "the" (ignoring upper/lowercase)
- Total number of times the word is "a" or "an" (ignoring case)

At the end of the input, the program prints

- Percentage of THE words is <<integer from 0 to 100>>
- Percentage of A or AN words is <<integer from 0 to 100>>

The hope is that these statistics uniquely characterize an author and can be used to find cases of plagiarism.

Please test your program well!

Reminder

Put useful comments into your code. Design and organize your code so that it is easy for others to understand.

Put your programs into a subdirectory called **Lab05** inside your **MyWork** directory, and remember to push your **Lab05** to GitHub before the deadline. This assignment must be turned in before Friday Sept 16th at 11:59pm.

Conclusion

In this lab you developed Java programs that are a bit more sophisticated than the earlier ones, relying on some of the design and coding experience you developed in CS110. These programs are a bit tricky, so feel free to contact the instructor and/or TA's with any questions.

Rubric

The first program is worth 5 points:

- One point if it has the correct name, is located in the correct directory, and compiles
- One point each if the output is correct for each of four test cases we will run

The other two programs are worth 20 points each:

- One point if it has the correct name, is located in the correct directory, and compiles.
- Three point each if the output is correct for each of the five test cases we will run.
- Zero to four points, based on the graders' judgment of the software design quality, software readability, comments, etc.